SYSTEM AND METHOD FOR INTERCEPTING PACKETS IN A PIPELINE NETWORK PROCESSOR

TECHNICAL FIELD OF THE INVENTION

The present invention relates in general to voice over Internet Protocol technology and more particularly to a system and method for intercepting packets in a pipeline network processor.

BACKGROUND OF THE INVENTION

Pipeline network processors are designed to forward Internet Protocol (IP) packets at an extremely high data rate in excess of two million packets per second. Pipeline network processors typically have a limited number of instruction cycles and memory to perform the applicable task. Implementing features or functions that do not fit the traditional packet forwarding model is a challenging endeavor and may have a negative impact on normal packet forwarding functions such as packet filtering and quality of service processing.

One of the seldom used features in a pipeline network processor is a wiretap or packet intercept function. Packet intercept provides support for a basic wiretap facility for voice over IP (VoIP) calls that is required by the United States Federal Communications Assistance for Law Enforcement Act. The wiretap facility is based on the Media Access Control (MAC) address of the customer premises equipment end user device. In a an environment that does not use a pipeline network processor, the source MAC address in the received packet is used to compare against the configured intercept MAC address list. For a packet being sent to a port, the packet's destination MAC address is used to compare against the configured intercept MAC address list. When a matching MAC address is found, a copy of the packet is encapsulated into a User Datagram Protocol (UDP) packet which is sent to a specified server at a given IP and port address.

Intercepting packets received from a port is a relatively simple operation in systems that use a central processing engine and not a pipeline network processor. The received packet contains the source MAC address

needed for the comparison. However, processing of a received packet in a pipeline network processor is not necessarily a simple operation due to the limited instruction cycle available. Moreover, intercepting packets sent to a port is extremely difficult in a pipeline network processor. The destination MAC address for the comparison is not in the packet but rather in the outbound Layer 2 encapsulation. The packet's payload is appended to the outbound Layer 2 encapsulation just prior to being forwarded to the network media. At this point, it is too late to perform additional processing in the pipeline network processor for MAC address comparisons. Without major restructuring of the packet forwarding path, interception cannot be accomplished in a pipeline network processor. Therefore, it is desirable to perform packet intercept processing in a pipeline network processor without requiring major restructuring of the packet forwarding path.

SUMMARY OF THE INVENTION

From the foregoing, it may be appreciated by those skilled in the art that a need has arisen for effectively intercepting packets in a pipeline network processor without adversely affecting the packet forwarding path. In accordance with the present invention, a system and method for intercepting packets in a pipeline network are provided that substantially eliminate or greatly reduce disadvantages and problems associated with conventional packet intercept techniques.

According to an embodiment of the present invention, there is provided a method of intercepting packets in a pipeline network processor that includes receiving an information packet from an inbound port. An outbound port for the information packet is then determined. A check is made to see if the outbound port has been identified for intercept processing. An identity of a destination is placed into the information packet. In response to the outbound port being identified for intercept processing, a determination is made as to whether the destination has been identified for intercept processing. A copy of the information packet is made in response to the destination being identified for intercept processing.

The present invention provides various technical advantages over conventional packet intercept techniques. For example, one technical advantage is to provide a packet intercept capability in a pipeline network processor. Another technical advantage is having minimal impact on normal packet forwarding operations and no restructuring of the pipeline network processor in order to provide the intercept feature. Yet another technical advantage is to avoid impacting the limited memory

resources in the pipeline network processor. Still another technical advantage is to provide an intercept feature for similar and dis-similar inbound and outbound network media. Other technical advantages may be readily ascertainable by those skilled in the art from the following figures, description, and claims.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention and the advantages thereof, reference is now made to the following description taken in conjunction with the accompanying drawings, wherein like reference numerals represent like parts, in which:

FIGURE 1 illustrates a simplified block diagram of a packet processing and distribution network;

FIGURE 2 illustrates a simplified block diagram of a pipeline network processor in the packet processing and distribution network;

FIGURE 3 illustrates a simplified logic flow diagram of packet intercept processing performed by the pipeline network processor;

FIGURE 4 illustrates a table used in the pipeline network processor to enable inbound packet interception;

FIGURE 5 illustrates a table used in the pipeline network processor to enable outbound packet interception;

FIGURE 6 illustrates a group index table and a configured intercept address table used by the pipeline network processor for address comparisons;

FIGURE 7 illustrates a simplified flow diagram of source and destination address comparisons of packets identified for intercept processing.

DETAILED DESCRIPTION OF THE INVENTION

FIGURE 1 is a simplified block diagram of a packet processing and distribution network 10. Network 10 includes an Internet Protocol network 12, a cable modem termination system 14, a cable modem unit 16, and a plurality of customer premises equipment 18. Cable modem termination system 14 includes a line card 20 to interface with Internet Protocol network 12, a line card 22 to interface with cable modem unit 16, and a packet processing unit 24.

FIGURE 2 is a simplified block diagram of packet processing unit 24. Packet processing unit 24 includes a route processor 30, a forwarding processor 32, and a plurality of interface units 34. Interface units 34 receive and transmit packets from and to line cards 20. Route processor 30 performs initializing of forward processor 32 and setting/updating of feature and forwarding tables in forwarding processor 32. Route processor 30 may process certain packets diverted from forwarding processor 32. Route processor 30 may also inject packets into forwarding processor 32 for transfer to interface units 34 and line cards 20. Among other functions, injected packets may provide results for diverted packets and route updates. Forwarding processor 32 is a pipeline network processor that receives packets, performs fast path switching, feature enforcement, and packet forwarding. Packets may be unicast having one destination or multicast having multiple destinations. Unicast and multicast packets follow separate forwarding paths through forwarding processor 32.

Forwarding processor 32 includes eight processing columns 36. Each processing column 36 has 128 instruction cycles to perform packet processing before

control is passed to the next processing column 36. If a processing column 36 does not complete its processing of a packet within the 128 instruction cycle, the packet is fed back to the processing column 36 for subsequent processing. Each processing column 36 includes a memory 38 to store tables used in the processing of a packet. Each memory 38 may be partitioned into a slow access memory and a fast access memory with varying amount of memory space.

Each processing column 36 performs a distinct processing function within forwarding processor 32. Column 0 performs initial identification and classification of a packet received from interface units 34. Column 1 performs a route lookup in order to properly forward the packet. Column 2 performs access control list processing to determine if the packet is allowed to be received and allowed to be forwarded. Column 3 performs input quality of service processing on the packet. Column 4 performs output quality of service processing on the packet and IP fragmentation if the packet length exceeds a desired output size. Column 5 performs MAC address rewriting. Column 6 performs output queuing of the packet. Column 7 performs output forwarding of the packet. For normal forwarding, a packet is identified and classified in Column 0 and a route for the packet is identified in Column 1. Input access control list processing is performed in Column 2 followed by input quality of service processing in Column 3. The packet returns to Column 2 for output access control list processing and is passed by Column 3 to Column 4 for output quality of service processing. Column 5 performs the MAC address rewrite and Column 6

places the packet in an output queue. Column 7 then forwards the packet towards its destination.

Forwarding processor 32 has a capability to perform a packet intercept function by comparing MAC addresses of packets received from upstream ports and sent to downstream ports. If a match is found, forwarding processor 32 diverts a copy of the originally received packet to route processor 30 for transfer to an intercept receiver. Alternatively, forwarding processor 32 may generate an intercept packet from the originally received packet for transfer to an intercept receiver. Inbound MAC address comparisons are performed after input access control list processing. Outbound MAC address comparisons are performed after output access control list processing. Since intercept processing is based on MAC addresses, interception of data or digitized voice packets may occur.

FIGURE 3 shows a flow diagram of the packet intercept process performed by forwarding processor 32. Initially, forwarding processor 32 receives a packet from a line card 20 through interface unit 34 at step 40. Forwarding processor 32 in Column 0 will determine whether the input port that the packet was received from is enabled for intercept processing.

Column 0 includes a table to indicate whether or not the packet intercept feature is enabled for an upstream port. FIGURE 4 shows an example of an input port configuration table indicating the enablement of the packet intercept feature. If enabled, an input intercept flag is set in the packet at step 42. The packet is then passed to Column 1 at step 44 to determine an outbound route for the packet. Forwarding processor 32 then

determines whether the outbound port has been enabled for intercept processing.

Column 1 includes a table to indicate whether the packet intercept feature is enabled for a downstream port. FIGURE 5 shows an example of an output port configuration table indicating the enablement of the packet intercept feature. If enabled, an output intercept flag is set in the packet at step 46 and the destination MAC address is stored in the packet. The packet then goes through input access control list processing in Column 2 at step 48.

After input access control list processing is performed, a determination is made as to whether the input intercept flag has been set at step 50. If not, the packet proceeds to input quality of service processing in Column 3 at step 52. If so, then a MAC address comparison is performed at step 54 to determine whether or not the packet is to be intercepted. Column 2 includes a table of configured intercept MAC addresses for each upstream and downstream port. Configured intercept MAC addresses may be grouped on a per port basis and accessed through an index table. An example of table indexing in Column 2 is shown in FIGURE 6. After the MAC address comparison, the packet is passed to input quality of service processing in Column 2 at step 52.

After input quality of service processing is performed, the packet is fed back to Column 2 for output access control list processing at step 56. After output access control list processing is performed, a determination is made at step 58 whether the output intercept flag has been set. If not, the packet is forwarded to output quality of service processing in Column 4 at step 60. If the output intercept flag is

set, the packet is passed on for MAC address comparison at step 62 before returning to output quality of service processing. After output quality of service processing, the packet is forwarded through Columns 5-7 for transfer from a line card 20.

FIGURE 7 shows the MAC address comparison performed by forwarding processor 32. Upon entering MAC address comparison, a determination is made at step 70 as to whether the packet has its input or output intercept flags set. For input intercept flags, a first buffer is loaded at step 72 with the source MAC address from the packet. A second buffer is loaded at step 74 with an intercept MAC address from a configured intercept MAC address table. A comparison of the first and second buffer is performed at step 76. If there is no match, a determination is made at step 78 as to whether there are more configured intercept MAC addresses in the table. If not, then the packet is forwarded to input quality of service processing at step 52. If so, the next configured intercept MAC address in the table is loaded into the second buffer at step 80 and the comparison of step 76 is performed again. This comparison loop will continue until there is a match or none of the configured intercept MAC addresses in the table match the source MAC address of the packet. If there is a match, a copy of the packet is diverted at step 82 to route processor for transfer to an intercept receiver or an intercept packet is generated by forwarding processor 32 for transfer to the intercept receiver as desired. The original packet is then further processed through forwarding processor 32. Forwarding processor 32 works similarly when the output intercept flag is set. The only difference is that the first buffer is loaded at step 84 with the

destination MAC address for the packet before the comparisons are performed.

The packet intercept feature may be implemented within forwarding processor 32 using inbound detection processing, outbound detection processing, common compare processing, and common final processing routines. TABLE 1 shows the performance of the inbound and outbound detection processing routines according to each processing column 36 in forwarding processor 32.

Column	Inbound Detection Processing	Outbound Detection Processing
0	Receive Packet from a Line Card **If Packet Intercept on vcci_in is ON** **Set PI_IN bit in Packet** PDONE(FIB - unicast) or PDONE (MFIB - multicast)	
1	Route Lookup (FIB or MFIB or TFIB) **If Packet Intercept on vcci_out is ON** **Set PI_OUT in Packet** PDONE(INPUT_ACL)	
2	Perform Input ACL **IF PI_IN is ON** **PDONE(PKT_INTERCEPT)** **Else** **PDONE(INPUT_QOS)**	Perform Output ACL **If PI_OUT is ON** **PDONE(PKT_INTERCEPT)** **Else** **PDONE(OUTPUT_QOS)**
3	PKT_INTERCEPT Path **Feedback Setup (pkt_type, pkt_direction)** **Save Context Bytes** PDONE(PKT_INTERCEPT)	PKT_INTERCEPT Path **Feedback Setup (pkt_type, pkt_direction)** **Save Context Bytes** PDONE(PKT_INTERCEPT)

Column	Inbound Detection Processing	Outbound Detection Processing
4	PKT_INTERCEPT Path Pass Packet PDONE(PKT_INTERCEPT)	PKT_INTERCEPT Path Pass Packet PDONE(PKT_INTERCEPT)
5	PKT_INTERCEPT Path **Copy Source MAC Address to compare area in Packet** PDONE(PKT_INTERCEPT)	PKT_INTERCEPT Path **Lookup Destination MAC Address** **Copy Destination MAC Address to compare area in Packet** PDONE(PKT_INTERCEPT)
6.	PKT_INTERCEPT Path Pass Packet PDONE(PKT_INTERCEPT)	PKT_INTERCEPT Path Pass Packet PDONE(PKT_INTERCEPT)
7.	PKT_INTERCEPT Path **Packet Moves from IPM to SDRAM - if not done yet** **Packet Feedback to Column- 0 on PKT_INTERCEPT Path**	PKT_INTERCEPT Path **Packet Moves from IPM to SDRAM if not done yet** **Packet Feedback to Column -0 on PKT_INTERCEPT Path**

TABLE 1

For the inbound detection processing routine, Column 0 receives a packet from source port 1 along a normal path and sets the input intercept flag in the packet if port 1 is enabled for packet intercept. The packet is then forwarded to Column 1 for route lookup to identify a destination port 2. Column 1 will also set the output intercept flag in the packet if port 2 is enabled for packet intercept. The packet is then passed to Column 2 where input access control list processing is performed, including any necessary feedbacks. Column 2 will check for the setting of the input intercept flag in the packet. If the flag has not been set, the packet will proceed to Column 3 along the normal path for input quality of service processing. If the flag has been set,

the packet will proceed along an intercept path to Column 3.

Along the intercept path, Column 3 performs feedback setup processing for the packet intercept feature. Feedback setup processing includes setting the packet type and packet direction. The packet direction is set to INBOUND. The packet type is set to either UNICAST or MULTICAST. Unicast packets have one destination while multicast packets have more than one destination. A number of packet bytes are saved in memory 38 of Column 3 since the packet intercept feature performs some overwriting of packet bytes. After feedback setup processing, the packet is transferred on the intercept path to Column 4 and then Column 5. Column 5 inserts a MAC address associated with the packet source into the packet. The packet continues along the intercept path through Column 6 to Column 7. At Column 7, the packet is fed back to Column 0 on the intercept path in order to perform the common compare processing.

Outbound detection processing may be entered for packets from input quality of service processing in Column 3 that do not have a set input intercept flag or that have already been intercept processed. Output access control list processing is then performed at Column 2 and a determination is made as to whether the output intercept flag is set. If not, the packet proceeds to Column 4 along the normal path for output quality of service processing. If the output intercept flag is set, the packet is forwarded along the intercept path to Column 3 for feedback setup processing where this time the packet direction is set to OUTBOUND. Certain packet bytes are also saved as discussed above. After feedback setup processing, the packet is transferred on

the intercept path to Column 4 and then Column 5. Column 5 determines and then inserts a destination MAC address associated with the packet destination into the packet. The packet continues along the intercept path through Column 6 to Column 7. At Column 7, the packet is fed back to Column 0 on the intercept path in order to perform the common compare processing.

TABLE 2 shows the performance of the common compare processing and final processing routines according to each processing column 36 in forwarding processor 32.

COLUMN	MAC Compare Processing	Final Processing
0	PKT_INTERCEPT Path Pass Packet PDONE(PKT_INTEREPT)	PKT_INTERCEPT_DONE Path Pass Packet PDONE(PKT_INTERCEPT_DONE)
1	PKT_INTERCEPT Path Pass Packet PDONE(PKT_INTERCEPT)	PKT_INTERCEPT_DONE Path **If bytes_restored is ON** **Restore Temp Flags** PDONE (PKT_INTERCEPT_DONE)
2.	PKT_INTERCEPT Path **Compare MAC Address(es)** **If found** **Feedback Update** **(intercepted = ON)** **COPY Packet Setup** **Else (not intercepted)** **Feedback Update** **(terminate = ON)** PDONE(PKT_INTERCEPT_ DONE)	PKT_INTERCEPT_DONE Path **If bytes_restored is ON** **Feedback Cleanup** **If direction INBOUND** **PDONE(INPUT_QOS)** **Else PDONE(OUTPUT_QOS)** **Else (intercepted packet)** **Feedback Update** **(terminate = ON)** PDONE(PKT_INTERCEPT_DONE)
3	PKT_INTERCEPT_DONE Path **If terminate is ON** **Restore Context Bytes** **Feedback Update** **(bytes_restored = ON)** PDONE(PKT_INTERCEPT _DONE)	INPUT_QOS Path Or OUTPUT_QOS Path Or PKT_INTERCEPT_DONE Path

COLUMN	MAC Compare Processing	Final Processing
4	PKT_INTERCEPT_DONE Path Pass Packet PDONE(PKT_INTERCEPT _DONE)	
5	PKT_INTERCEPT_DONE Path Pass Packet PDONE(PKT_INTERCEPT _DONE)	
6	PKT_INTERCEPT_DONE Path Pass Packet PDONE(PKT_INTERCEPT _DONE)	
7	**Packet copied and copy diverted to intercept receiver if INTERCEPTED Packet Feedback to Column-0 on PKT_INTERCEPT_DONE Path**	

TABLE 2

Common compare processing is performed in Column 2 on a feedback pass along the intercept path from either inbound detection processing or outbound detection processing. The packet passes through Columns 0 and 1 to Column 2 where the MAC address inserted into the packet is compared to the configured intercept MAC addresses in the table maintained in Column 2. If no match is found, a terminate flag is set, the packet is forwarded along an intercept complete path to Column 3 where the saved bytes are restored in the packet, and the packet is transferred through Columns 4-7 to common final processing. If a match is found, an intercepted flag is set, divert setup processing is performed, and the packet is forwarded along the intercept complete path through Columns 3-6 to Column 7 where a copy of the original packet is made and either diverted to route processor 30 or sent directly to the intercept receiver. the packet is then fed back on

the intercept complete path to Column 0 for common final processing.

Common final processing is performed on a feedback pass from common compare processing. Common final processing is performed in one pass for packets which were not intercepted and in two passes for packets which were intercepted. At final processing, the packet is sent through Column 0 to Column 1 along the intercept complete path where a check is made to see if bytes were restored in the packet. If so, indicating that the packet was not intercepted, other flags are restored and the packet proceeds along the intercept complete path to Column 2 where feedback cleanup is performed. If the packet direction is INBOUND, the packet is then transferred on the normal path to input quality of service processing in Column 3. If the packet direction is OUTBOUND, the packet is then transferred along the normal path to output quality of service processing in Column 3. If packet bytes have not been restored, indicating that the packet was intercepted, the terminate flag is set in Column 2 and the bytes are restored in Column 3. The packet continues along the intercept complete path through Columns 4-7 and fed back to Column 0. In this last feedback path, the packet goes to either input or output quality of service processing as described above.

Forwarding processor 32 is able to intercept packets, especially outbound packets, without restructuring the pipeline network processor implementation or severely impacting other normal forwarding features. Forwarding processor 32 performs a look ahead into the outbound re-encapsulation Layer 2 header to find the needed destination MAC address for

storage and subsequent comparison to reference destinations in an intercept list. The packet intercept technique in saving the MAC address in the packet for comparison by forwarding processor 32 operates regardless of whether the packet is received and sent over the same or different inbound and outbound media types. Thus, for example, forwarding processor 32 can handle packets to and from inbound and outbound Ethernet links or to and from inbound ATM and outbound Ethernet links and vice versa.

Unicast or multicast packets that are dropped during normal processing in forwarding processor 32 may be excluded from intercept processing. A packet is dropped if the format of the packet has problems or if the route cannot be determined. These packets may be diverted to route processor 30 for further handling. Unicast and multicast packets that are normally diverted to route processor 30 or injected therefrom may also be excluded from intercept processing to avoid sending and receiving duplicate copies of packets between route processor 30 and forwarding processor 32. For these exclusions, route processor 30 may be configured to handle intercept processing unless it is desired to have forwarding processor 32 provide intercept processing for these packets.

Though forwarding processor 32 may forward the copy of the packet upon detecting interception to route processor 30 for transfer to an intercept receiver, it may be desirable to limit the functions of route processor 30 to the handling of configuration, network management functions, and route updates. Moreover, the volume/rate of traffic which may be sent from forwarding processor 32 to route processor 30 is smaller than the

volume/rate of traffic which forwarding processor 32 can send and receive. Thus, having the forwarding processor copy and forward the copied packet to the intercept receiver will lessen the load of route processor 30. Forwarding processor 32 is still free to perform normal packet forwarding.

Thus, it is apparent that there has been provided, in accordance with the present invention, a system and method for intercepting packets in a pipeline network processor that satisfies the advantages set forth above. Although the present invention has been described in detail, it should be understood that various changes, substitutions, and alterations may be made herein. For example, though discussed in relation to a cable modem and cable ports, the present invention may equally be applicable in other packet distribution environments. Also, though discussed with respect to MAC address comparisons, the present invention may equally apply to the use of other addressing schemes including Internet Protocol addresses. Other examples may be readily ascertainable by those skilled in the art and may be made herein without departing from the spirit and scope of the present invention as defined by the following claims.